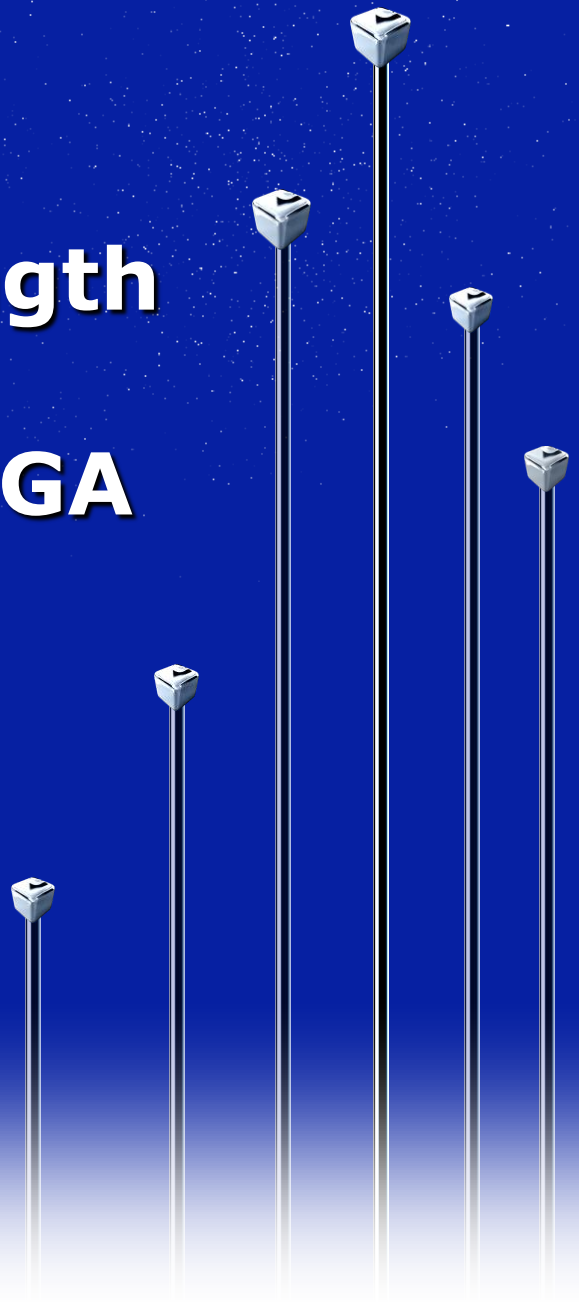
4-Length and 8-Length FFT Calculator Implemented in FPGA

By
Su Yu





Contents

- Introduction
- Objective
- Implementation
- Result



Introduction

- A fast Fourier transform (FFT) is an efficient algorithm to compute the discrete Fourier transform (DFT) and its inverse, which is widely used in signal processing and related fields to analyze the frequencies contained in a sampled signal.

- The Cooley–Tukey algorithm, named after J.W. Cooley and John Tukey, is the most common fast Fourier transform (FFT) algorithm.

Introduction

- A radix-2 FFT is the simplest and most common form of the Cooley–Tukey algorithm. It divides a DFT of size N into two interleaved DFTs (hence the name "radix-2") of size N/2.

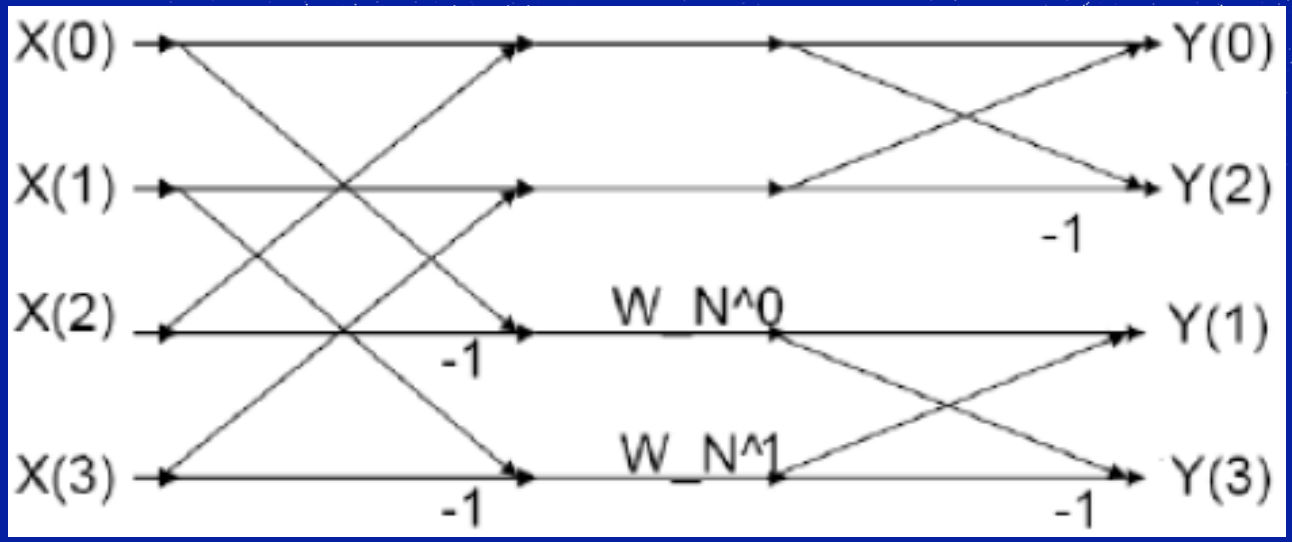
$$Y(k)=\sum_{n=0}^{N-1}x(n)W_N^{kn};\quad k=0,1,\ldots,N-1;\quad W=e^{-j\frac{2\pi}{N}};\quad N=even;$$

$$For\quad k\quad is\quad even,\quad Y(2r)=\sum_{n=0}^{\frac{N}{2}-1}[x(n)+x(n+\frac{N}{2})]W_{\frac{N}{2}}^{rn};$$

$$For\quad k\quad is\quad odd,\quad Y(2r+1)=\sum_{n=0}^{\frac{N}{2}-1}[x(n)-x(n+\frac{N}{2})]W_N^nW_{\frac{N}{2}}^{rn};$$

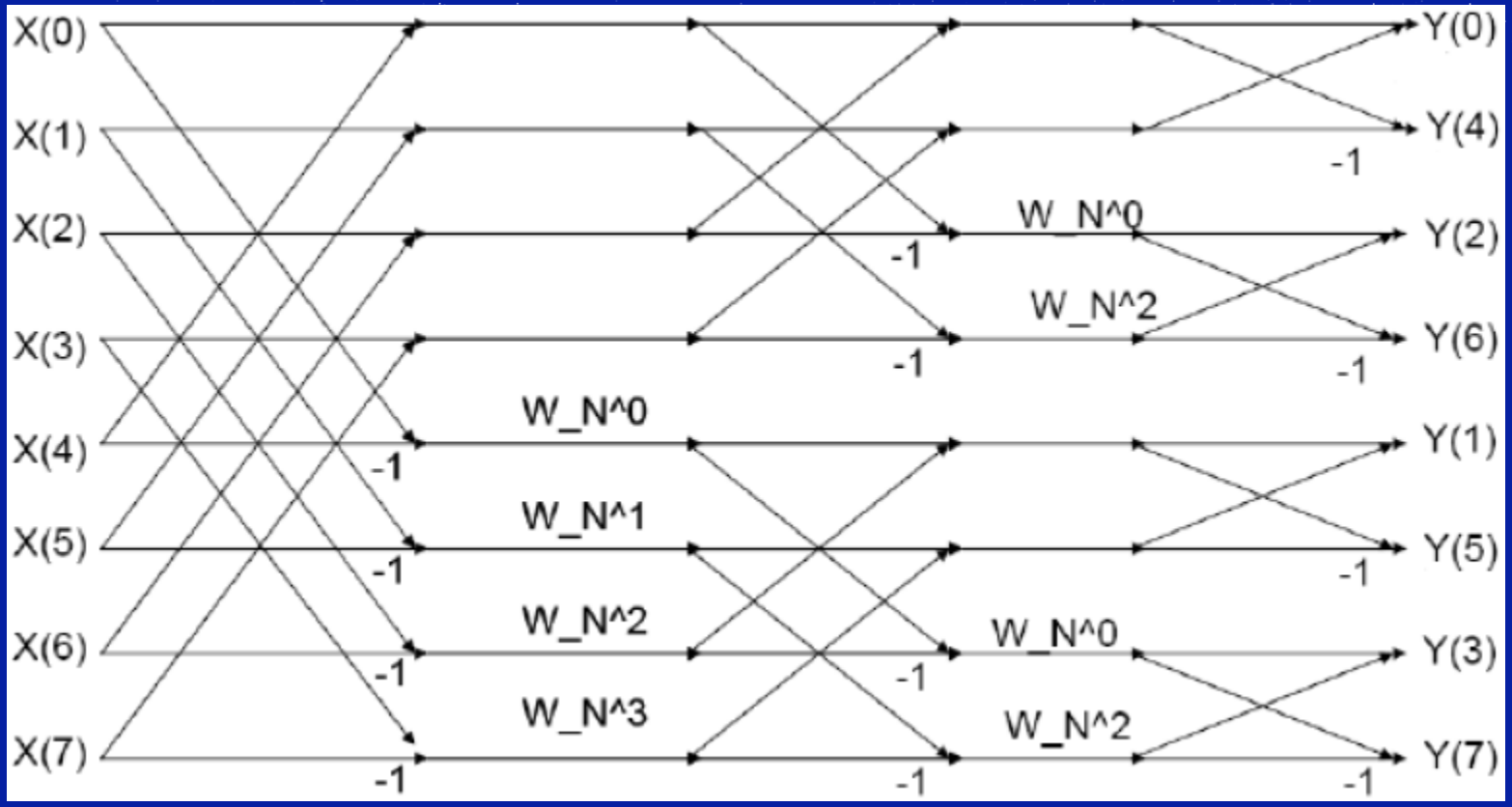
Introduction

- For 4-Length FFT:



Introduction

- For 8-Length FFT:





Contents

- Introduction
- Objective
- Implementation
- Result



Objective

- This project realizes the radix-2 Cooley-Tukey algorithm for 4-length and 8-length FFT calculator in the ALTERA DE2-70 FPGA board;
- The 4-length or 8-length could be selected by user;
- The input sequence could be given by user;
- The calculation result is shown on 7-segment numeric LEDs.

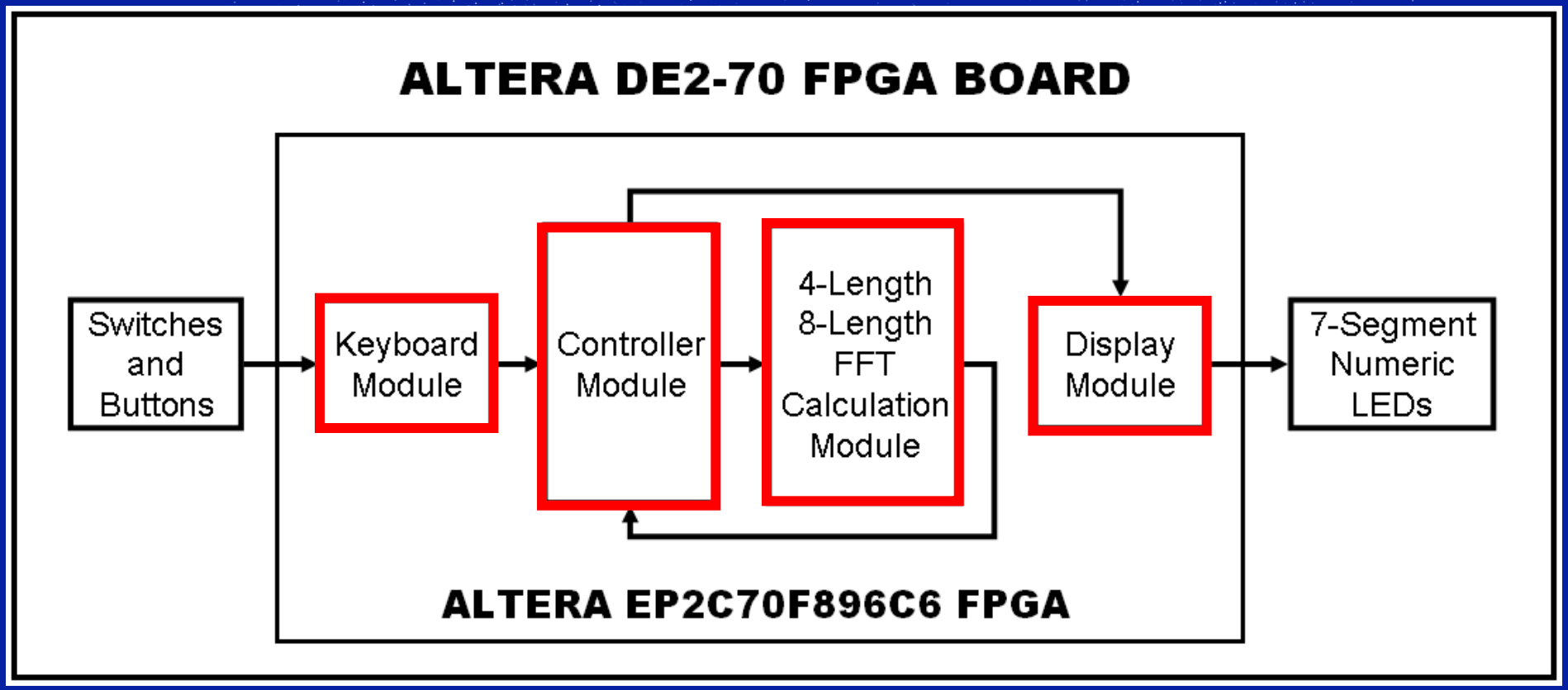


Contents

- Introduction
- Objective
- Implementation
- Result

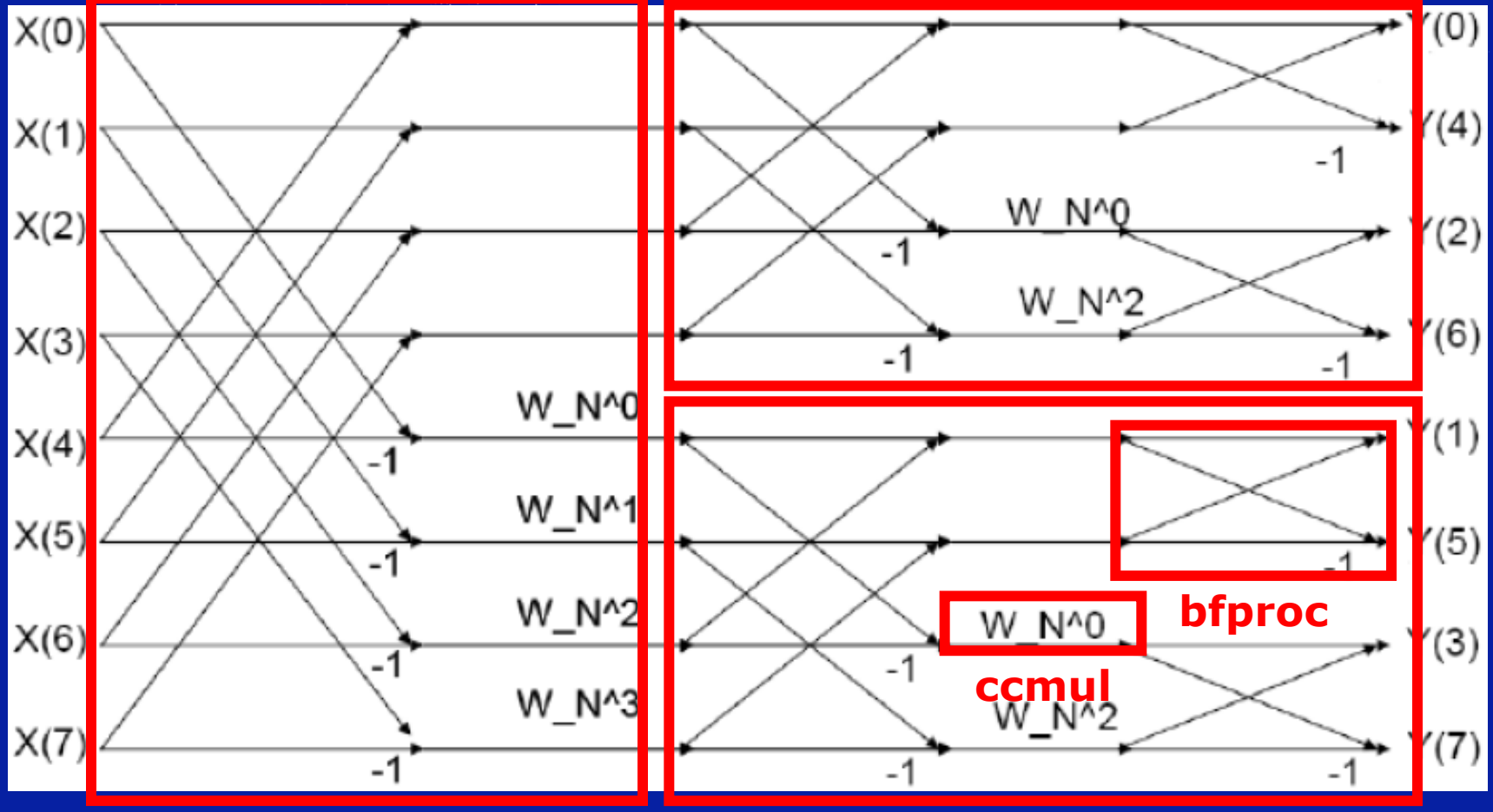
Implementation

- System block diagram:



Implementation

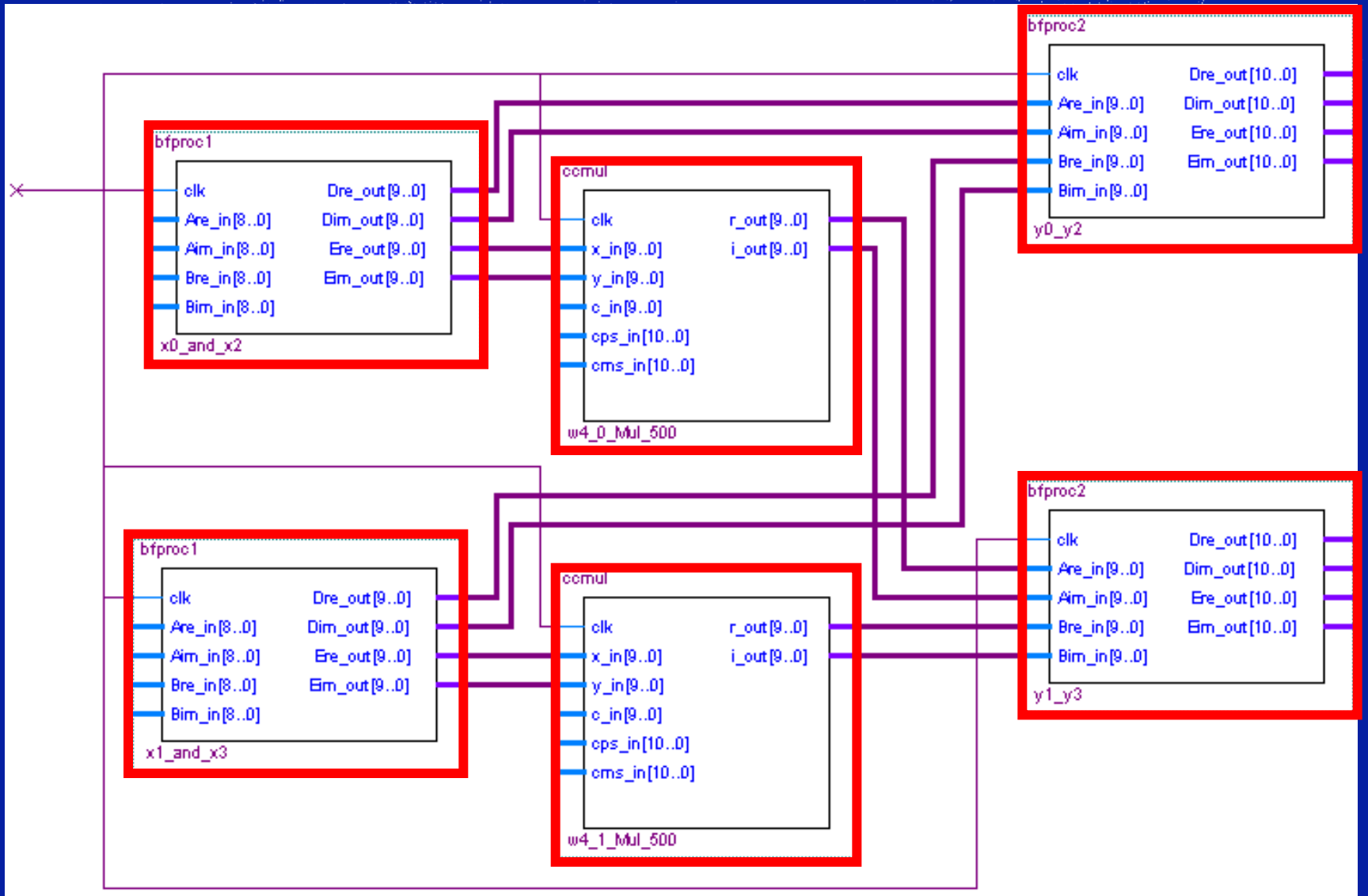
- 4-Length and 8-Length FFT Calculation Module:



stage1_8fft calculator_4fft

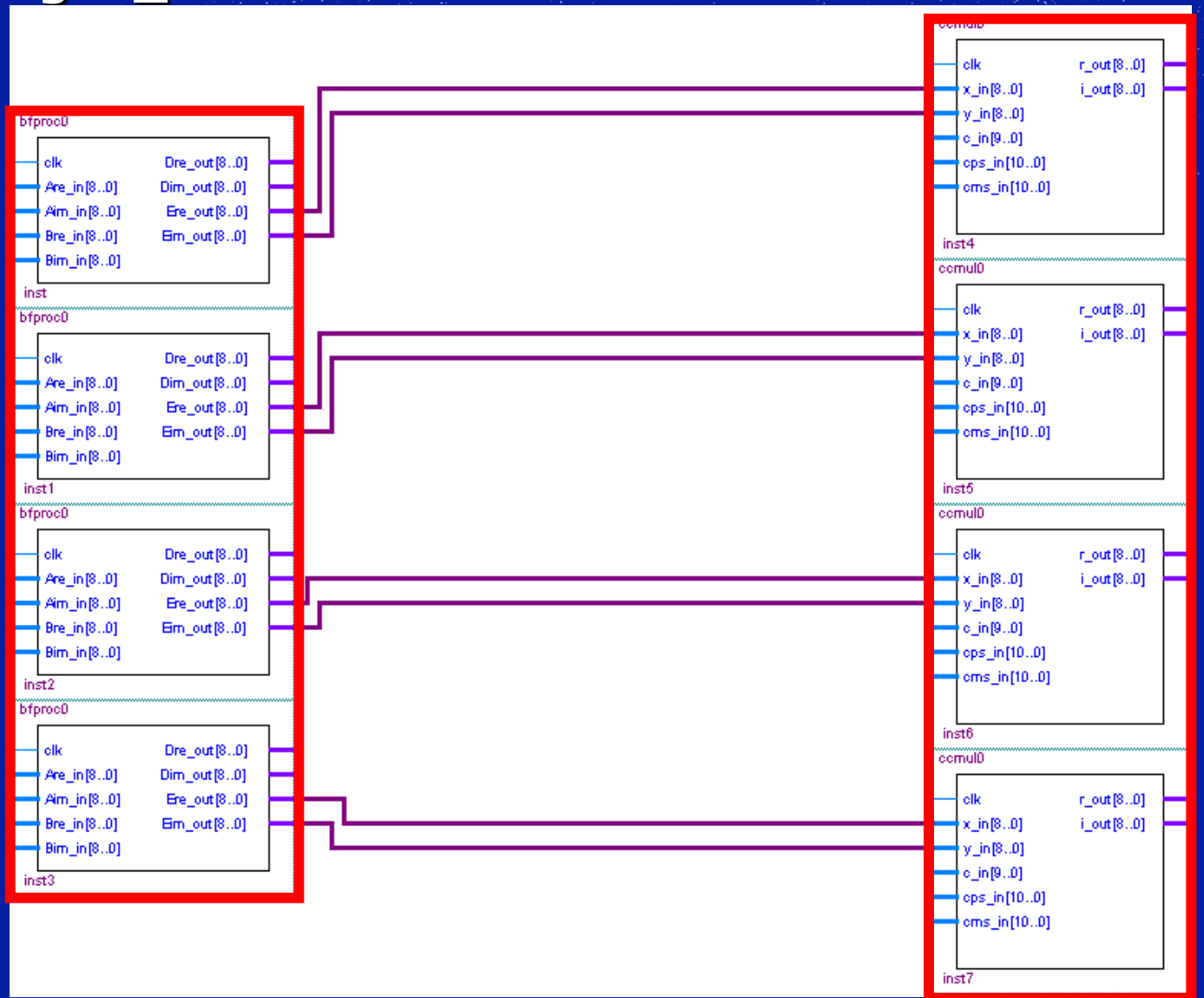
Implementation

- calculator_4fft:



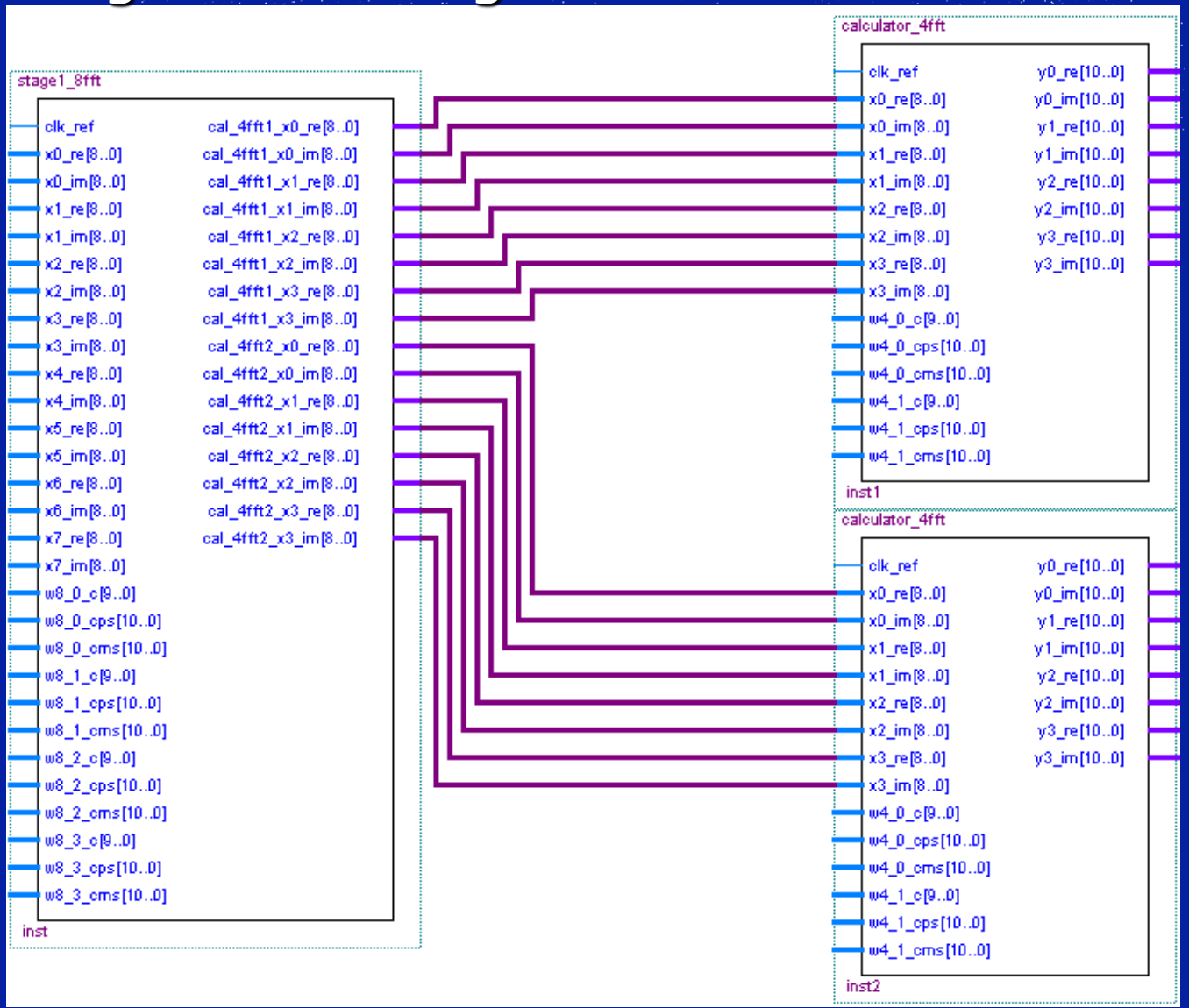
Implementation

- stage1_8fft:



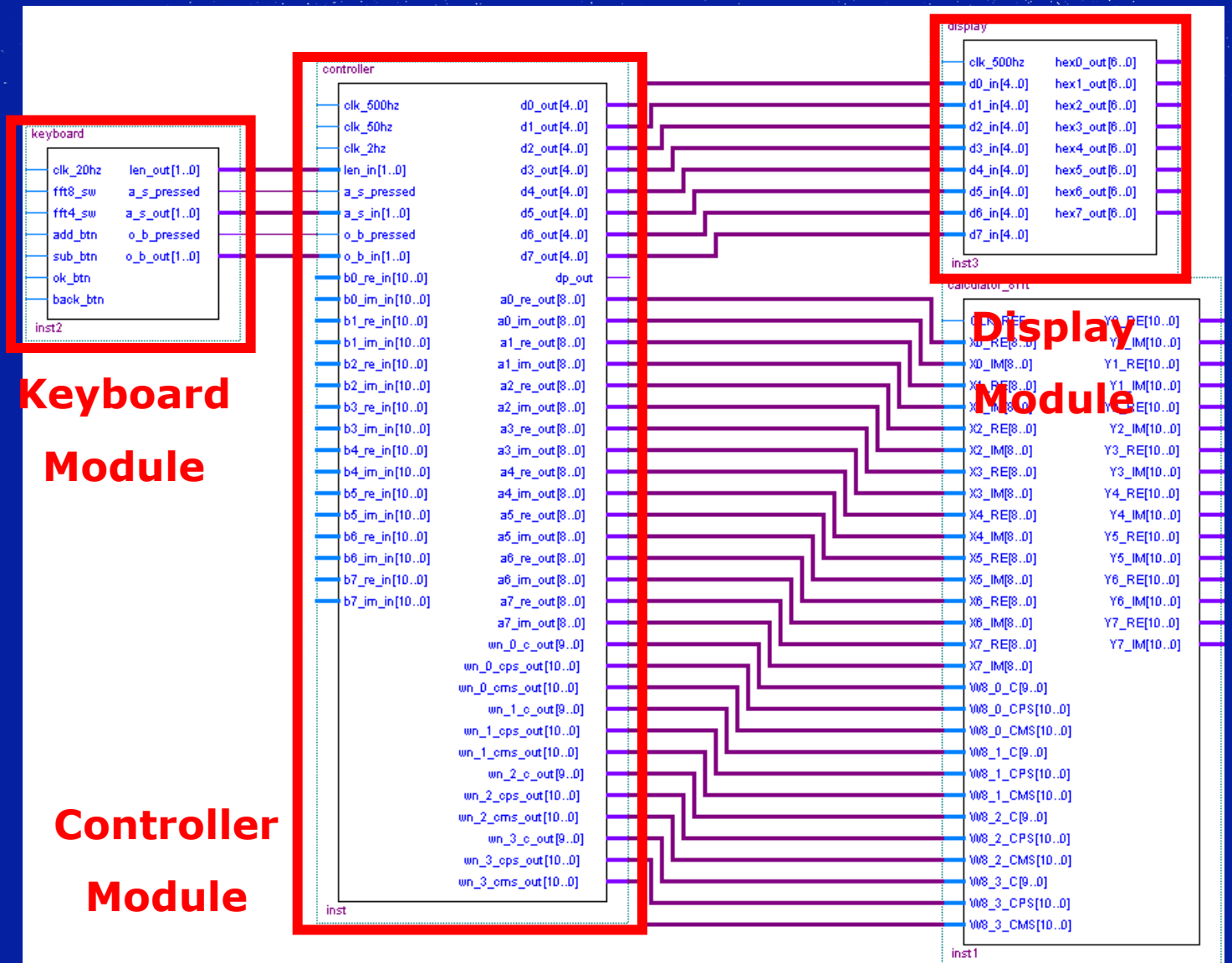
Implementation

4-Length and 8-Length FFT Calculation Module:



Implementation

- Other modules:





Contents

- Introduction
- Objective
- Implementation
- Result

Result

- To test 8-length FFT:

```
>> a=[3, -2.5+2.3i, 0.5-1.5i, 3.5-3.5i, -5-4i, 1.3+3.2i, -0.8+1.1i, -3.7i]

a =

  Columns 1 through 6

   3.0000            -2.5000 + 2.3000i   0.5000 - 1.5000i   3.5000 - 3.5000i  -5.0000 - 4.0000i   1.3000 + 3.2000i

  Columns 7 through 8

  -0.8000 + 1.1000i         0 - 3.7000i

>> fft(a)

ans =

  Columns 1 through 6

         0 - 6.1000i  -0.2569 + 2.1343i  11.0000 + 1.1000i  15.2669 + 6.2899i  -4.6000 - 2.7000i  11.0569 + 3.2657i

  Columns 7 through 8

 -14.4000 - 8.3000i   5.9331 + 4.3101i
```



Result

- To test 4-length FFT:

```
>> a=[12.7-3.4i, -8.0+1.2i, -15.9-6.3i, -3.0+9.5i]

a =

  12.7000 - 3.4000i  -8.0000 + 1.2000i -15.9000 - 6.3000i  -3.0000 + 9.5000i

>> fft(a)

ans =

 -14.2000 + 1.0000i  20.3000 + 7.9000i   7.8000 -20.4000i  36.9000 - 2.1000i
```

REFERENCES

- Ali N. Akansu and Mustafa U. Torun, LABORATORY MANUAL ECE 641 Laboratory for Digital Signal Processing with Field Programmable Gate Arrays (FPGA);
- ALTERA, DE2-70 Development and Education Board User Manual. Version 1.08, Copyright ©2009 Terasic Technologies;
- U. Meyer-Baese, Digital Signal Processing with Field Programmable Gate Arrays. 3rd Edition, Springer, 2007;
- http://en.wikipedia.org.

Thank you!

